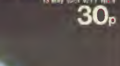
POPULAR
Computing
WEEKLY

13 May 1982 Vol 1 No 4

30p



Mind Reader analyses you

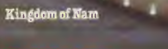
Reviews: Aliens



Bolton port Vic joysticks

Chaining ZX81 programs

Kingdom of Nam



Win ZX81 software


BYTESHOP COMPUTERLAND
VIC
VIC 20

POPULAR Computing WEEKLY

The Team

Editor
Duncan Scot

Production consultants
Myles Hewitt
Eric Robbie

Editorial Secretary
Fiona McCormick

Advertisement Manager
David Lake [01-839 2846]

Advertisement Executive
Peter Chandler [01-839 1881]

Software Editor
Peter Gerrard [01-839 1855]

Publishing Director
Nick Hampshire

Popular Computing Weekly,
Hobhouse Court,
19 Whitcomb Street,
London WC2 7HF
01-839 6835 and 01-839 1855

Published by Sunshine Publications Ltd, Hobhouse Court 19 Whitcomb Street, London WC2 7HF

Typesetting, origination and printing by Chesham Press Chesham, Bucks

Distributed by S M Distribution London SW9 01-274 8611 Telex. 261643

Subscriptions
You can have *Popular Computing Weekly* sent to your home the subscription rate is £19 95 per year for addresses in the UK

How to submit articles
Articles which are submitted for publication should not be more than 1000 words long

All submissions should be typed and a double space should be left between each line

Programs should whenever possible be computer printed

At present we cannot guarantee to return every submitted article, so please keep a copy

Accuracy
Popular Computing Weekly cannot accept any responsibility for any errors in programs we publish although we will always try our best to make sure programs work

This Week


Cover illustration by Stuart Hughes

Editorial

Two weeks ago we suggested that Vic-20 owners may be paying too much for their peripherals The reason was that competition in the market would soon bring the prices down

As we report on the News page it is rumoured that the launch of the ZX Spectrum by Sinclair has so stunned Commodore that the much heralded launch of the Ultimax/Vic-10 has now been abandoned

Commodore had only released the final version and specification of the Ultimax the day before the launch of the Spectrum For a company the size of Commodore to react that quickly to the launch of a competitive machine would certainly be surprising

It could be that Commodore has made a mistake in that the high-resolution and game playing features of the Vic-10 would have appealed to a different section of the market anyway

Commodore now looks very vulnerable Over the next couple of months it will need to rethink the whole of its marketing strategy

Next Week



Journey with us into the science of sound Learn how to manipulate the music of the spheres, in our super sonic issue

Sinclair boosts software

Coinciding with its launch of the ZX Spectrum, Sinclair has announced a new range of software for the ZX81. Other announcements including news that the price of the ZX 16K RAM pack has been reduced from £49.95 to £29.95. At the same time the price of the ZX printer, over which Sinclair has a monopoly supply, has been increased to £59.95 from £49.95.

The nature of the announcements about the software, RAM packs and the ZX printer implies that Sinclair is no longer keen on encouraging the development of independent suppliers for the ZX market, many of whom have had their marketing plans thrown into confusion by the reduced prices.

The new software has been developed for Sinclair partly by ICL and partly by a software house called Psion. Mail-order sales of the new cassettes will begin in the UK in May. Part of the range will also be sold through W H Smith's stores.

Eight cassettes together form the *Fun to Learn* series and are each available at £6.95. The series comprises English Literature I and II, Geography, History, Mathematics, Inventions, Spelling and Music.

A further eight cassettes, retailing at £4.95 from the *Super Programs* series, which contains games, quiz, conversion and household programs.

ICL's Collector Pack provides a program enabling collectors, of, for example, stamps or coins, to hold a maximum of 400 records of up to six items on one cassette. The Club Record Controller is written to hold the personal records of 100 people. Both are available at £9.95.

The final cassette from ICL, available at £6.95, offers an explanation of the fashionable subject of Biorhythms.

Seven Psion cassettes range in price from £3.95 to £7.95 and include a Backgammon game, a six level Chess program, Vu-Calc, which constructs, generates and calculates large tables for applications such as final analysis budget sheets and projections, Vu-File for general purpose filing and information retrieval, Flight Simulation, Space Raiders and Bombers and Fantasy Games.

All the new cassettes require the use of the add-on 16K RAM pack with the exception of the ICL *Super Programs* which need only 1K.

Sinclair claims that a reduction in the costs of chips have enabled him to reduce the price of the RAM pack to £29.95 and that rising production costs have forced the increase in the price of the printer to £59.95.

Both these claims have been met with some scepticism by the Sinclair add-on industry which believes that Sinclair is trying to strengthen his monopoly hold on the printer market while undermining the profits of the add-on memory suppliers.

There are however, strong rumours that a rival to the ZX Printer will be launched in the early autumn.

Sinclair hopes to boost the ZX81 while promoting Spectrum

The new Sharp personal computer with 48K RAM. The MZ80A has a standard keyboard, green CRT display and a cassette deck in one self-contained unit. The basic price is £549. More details are available from Sharp on 061 205 2333

Stack your RAM in the Storeboard

A new idea in Vic peripherals is the Storeboard from Stack Computer Services of Liverpool. It is a boxed printed circuit board which can take up to 27K of RAM fitted internally beneath a detachable cover.

The unit slots into the memory expansion port on the Vic, and stands at the same level. Using gold edge connectors it ensures good contact with the Vic and a further point in its favour is that it doesn't require any extra power when in use.

The Storeboard costs £49 plus VAT, and is available from any Vic dealer.

Confusion over Commodore counter-attack

Speculation is mounting about how Commodore will react to the launch of the ZX Spectrum. The first story we heard from a Commodore mole was that the newly announced Ultimax/Vic-10 computer was to be abandoned and that the price of the new Vic-30 would be reduced to £125 to match that of the Spectrum.

According to this version of the internal wranglings as Commodore, the decision to abandon its long-held plans for the home computer market came after a 15-page technical report on the Sinclair Spectrum was presented to Commodore marketing manager John Baxter.

The original scheduled delivery date for the Vic 30 was around September this year.

Commodore's spokesman Peter Walker denies that any such decision has been taken. He says that Commodore is still considering whether to change the specifications or prices of any of its new computers.

One major Commodore dealer claims that he has been assured by Commodore that the Ultimax/Vic-10 will be kept alive and that the price of the Vic 30 will only be reduced to £160.

The Vic-30 has a screen resolution of 320 × 200 pixels and individual pixel movement. There are 16 colours which can be displayed on the screen at any one time, a true music synthesiser on board, 16K of RAM and a full typewriter keyboard. In external appearance the computer will look identical to the Vic-20.

If Commodore significantly reduces the price of the Vic-30, it will also need to bring forward the scheduled delivery dates to stand a chance of offering serious competition to Sinclair.

The similarity between the specifications of the Vic-30 and the ZX Spectrum is striking. Sinclair however, has the advantage of a captive market of ZX81 users, the ZX printer and the promise of the Spectrum RS232 link.

Club Reports

Is your club involved in any special projects? Use this page to tell the world about it.

You need never be alone with a Laserbug

***Trevor Sharples,** a founder of Laserbug, explains the aims of this BBC Micro user group*

When the first information about the BBC Microcomputer was leaked to the outside world a great many people began to take interest. A project commissioned by the BBC would surely be a winner on a national scale.

This was confirmed by initial 'hands-on' experience. It was then that the need for a user group became apparent. The BBC Micro was not just another computer — it was five micros rolled into one. Users would need to help each other if they wanted to get all they could out of the computer.

A whole series of ideas were thrashed around by the few of us caught up at the start. What should a user group do? How should we set about it?

We eventually settled on our 'constitution'. To put it very simply: The BBC user group should be the sort of user group that any one of us would have no hesitation in joining.

First and foremost we needed a name. Because the BBC had reservations about 'The BBC User Group' or 'The National BBC User Group' those were definitely out. We needed a dynamic-sounding acronym that people were going to remember. A combination of chance and playing around with anagrams resulted in LASERBUG — London and South East Region BBC Microcomputer User Group.

So Laserbug was born. We had a user group, but what of our policies?

Soft scribblers and hard freaks

We had already decided to ensure that Laserbug would exist solely for the support and enhancement of the BBC Micro. We also agreed that Laserbug should be flexible. The user group should go in the direction (or directions) that its members wanted it to go. But who would those members be?

The answer was simple — anyone who owned or who had access to a BBC Microcomputer. There would be complete beginners and others with varying degrees of experience. There would be the enthusiast and 'I've - got - one - because - I - ought - to - know - about - computers' person. There would be the games addict and the home businessman. There would be the software scribbler and the hardware freak. We wanted Laserbug to satisfy them all.

Laserbug wants to encourage the exchange of ideas, says Trevor Sharples

We want to encourage the exchange of ideas and discoveries, and a user group is the ideal way of setting about achieving that object. We would like to see the members telling us what they want out of Laserbug — because Laserbug is their user group.

A lot of inter-member communication will be relayed by the Laserbug Newsletter. The newsletter is a monthly 16-pager that is distributed to the user group members. It follows the established format of computer magazines in having sections on news, reviews, program listings, letters, projects, competitions, hints and tips, and whatever else comes our way that is worth printing.

The Laserbug newsletter provides a nationwide forum for information exchange, but that, we feel, should not be all that a user group offers. The need for meetings between BBC Microcomputer owners is recognised as being both essential for the development of the BBC Micro user and as being intrinsic to the running of a user group.

We intend to hold a series of meetings for all user group members (and interested non-members) to attend, but these will be infrequent and cause problems for some people to attend. Distance may be an obstacle as most meetings will be in London.

So a series of smaller, local meetings seems to be the answer. It is our aim, as Laserbug, to organise, or help organise, a national network of local user groups. It could be as small as a group of half-a-dozen in someone's front room, or a larger affair at the local church hall.

Belonging is the best way

We feel very strongly that belonging to a local user group is the best way to get the most out of your BBC Micro. We would like anyone coming into possession of a BBC Micro to be able to get together with other owner/users wherever he lives.

Membership of Laserbug costs £12 per year. That includes 12 issues of the newsletter, free entry to our meetings, exhibitions and any other activity we dream up. Or £1 will buy a sample copy of the newsletter if you send it with a large (9in × 15in) SAE (16½p stamp) to: Laserbug, 4 Station Road, Woodgrange Road, London E7 0NF.

Write to Club Reports, *Popular Computing Weekly*, Hobhouse Court, 19 Whitcomb Street, London WC2 7HF with details of successes you have had with your club, with ideas for helping clubs along and with any news of special meetings. We look forward to hearing from you.

COVER STORY

Mind

Are you worried about the way your life is going?

Don't waste your money on a 'shrink'

Use your ZX81 and this easy-to-run program to provide the solution to all your psychological problems.

Mind Reader has been specially developed for *Popular Computing Weekly* by **Dave Middleton.**

Mind Reader was originally developed as an example of pseudo-artificial intelligence and it exhibits a far more intelligent response than programs 10 times larger

The main drawback is that the sense of the program's response depends on the intelligence shown by the user when making entries If the user types in garbage then *Mind Reader* replies with garbage A true artificial intelligence program requires a massive database and needs a very fast computer to analyse the data and give a response to the user in reasonable time

Even *Mind Reader* gets a bit slow on the ZX81 when a long sentence has to be checked

This version of *Mind Reader* is about as simple as it can be The program is waiting for the user to enter a response based around personal references, for example, 'I or me

If the sentence does not contain a personal reference then *Mind Reader* prompts the user to make one This process is repeated three times and if there is still no response then *Mind Reader* gives a sharp response and asks for another subject

When a response is made that contains a personal reference then *Mind Reader* will change it from first person to second person, for example 'I am becomes you are' By changing first person to second person the user's response can be directed back as a question, in this case — 'Why do you think you '

The program

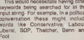
Most of the work in the program is accomplished from line 5000 onwards

A$: the input from the user
B$: A$ is converted into B$ with all first person words being changed to second person
C: the position of the conversion within A$
L: the number of characters which have been converted from first to second person
N: conversion type
0 — no personal references, eg, It was nice
1 — personal reference, eg 'I', 'me'
2 — reference to *Mind Reader*, eg you, your
NV accepts the highest value of N over the complete length of A$

The control structures for the prompts are very simple in this example program and non-specific in their actions

The program could be made a great deal more complex so that *Mind Reader* could ask questions on a specific topic — political, religious or sexual views, perhaps

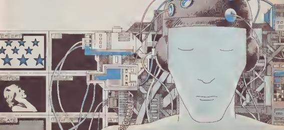
This would necessitate having other keywords being searched for in the input string For example, in a political conversation these might include words like Conservative Labour, Liberal, SDP, Thatcher, Benn and Foot





Reader

```
10 REM MINDREADER
20 REM BY DAVE MIDDLETON
30 REM
998 REM FR.MPT.
999 REM
1000 CLS
1010 PRINT "WHAT IS YOUR FIRST NAME
1020 INPUT N$
1030 PRINT N$
1040 PRINT "WHAT SUBJECT WOULD YOU LIKE"
1050 PRINT "TO TALK ABOUT   N$
1060 INPUT S$
1070 PRINT S$
1080 PRINT N$ "  WHAT WOULD YOU LIKE TO
1090 PRINT "SAY ABOUT   S$
1100 GOSUB 5000
1120 IF NV>0 THEN GOTO 2000
1130 PRINT "WHAT ELSE WOULD YOU LIKE TO SAY"
1140 PRINT "ABOUT " S$ "  " N$
1160 GOSUB 5000
1200 IF NV>0 THEN GOTO 2000
1210 PRINT "THAT IS INTERESTING " N$   BUT"
1220 PRINT "WHAT ARE YOUR PERSONAL VIEWS"
1240 GOSUB 5000
1300 IF NV>0 THEN GOTO 2000
1310 PRINT N$ "  PLEASE BE MORE PERSONAL
1320 GOSUB 5000
1400 IF NV 0 THEN GOTO 2000
1410 PRINT S$ " IS A SUBJECT WHICH YOU"
1420 PRINT "SEEM UNABLE TO TALK ABOUT AT
1430 PRINT "A PERSONAL LEVEL   LET US TRY ANOTHER    N$
1460 GOTO 1040
2000 IF NV 1 THEN GOTO 3000
2010 PRINT "WHY DO YOU THINK
2020 IF B$( 1 TO 3) > YOU" THEN PRINT  YOU "
2030 PRINT B$
2040 GOTO 1100
3000 PRINT "WE ARE TRYING TO FIND OUT YOUR"
3010 PRINT "VIEW  ABOUT   S$ "+NOT MINE"
3020 GOTO 1100
4998 REM ANALYSE STATEMENT
4999 REM
5000 INPUT A$
5005 PRINT A$
5007 FAST
5008 LET LE=LEN A$
5010 LET A$=A$+"
5020 LET B$="
5030 LET C=1
5040 LET NV=0
5050 LET N=0
5060 IF A$(1 TO 3)    BYE  THEN GOTO 5100
5070 PRINT "I ENJOYED TALKING TO YOU " N$
5080 GOTO 9999
5100 IF A$(C TO C+4)<> I AM " THEN GOTO 5200
5110 LET B$=B$+"YOU ARE "
5120 LET L=5
5130 LET N=1
5140 GOTO 6100
5200 IF A$(C TO C+2)<>"ME   THEN GOTO 5300
5210 LET L=3
5230 LET N=1
5260 GOTO 6100
5300 IF A$(C TO C+7)<>"YOU ARE   THEN GOTO 5400
5310 LET B$=B$+"I AM "
5320 LET L=8
5330 LET N=2
5360 GOTO 6100
5400 IF A$(C TO C+4)<> YOUR   THEN GOTO 5500
5410 LET B$=B$+"MY
5420 LET L=5
5430 LET N=2
5440 GOTO 6100
5500 IF A$(C TO C+2)<> MY " THEN GOTO 5600
5510 LET B$=B$+ YOUR
5520 LET L=3
5530 LET N=1
5540 GOTO 6100
5600 IF A$(C TO C+3)"  YOU " OR NV=1 THEN GOTO 5700
5610 LET B$=B$+ I "
5620 LET L=4
5630 LET N=2
5640 GOTO 6100
5700 IF A$(C TO C+3)   YOU   OR NV=0 THEN GOTO 5800
5710 LET B$=B$+ ME "
5720 LET L=4
5730 LET N=2
5740 GOTO 6100
5800 IF A$(C TO C+1)<>"I " THEN GOTO 5900
5810 LET B$=B$+"YOU
5820 LET L=2
5830 LET N=1
5840 GOTO 6100
5900 IF A$(C TO C+2)="WE " THEN LET N=1
5990 IF A$(C TO C+2)="US  THEN LET N=1
6000 LET B$=B$+A$(C TO C)
6010 LET L=1
6100 IF N>NV THEN LET NV=N
6110 LET C=C+L
6120 IF C<=LE THEN GOTO 5100
6125 SLOW
6130 RETURN
```



Reviews
software

Alien

Available from any Commodore Vic dealer
Price £19.95

Like many of the games cartridges available for the Vic, *Alien* bears remarkable resemblance to an existing arcade game. It's a fairly accurate reproduction as well, making good use of the Vic's programmable characters.

And so to the game itself. This is activated simply by plugging the cartridge into the back of the Vic and after an initial period of centring the image on the screen (why does this always appear to be the case? you need to play with the cursor keys in order to get the picture centred up) you're ready to go. You control a little man who runs around a maze, being chased by (at first) four alien beings. The idea is to kill the aliens off, and this is where the charm of the game becomes apparent.

Rather than the usual space war scenario, where you're merrily blasting down spaceships raining down on you from above, this one is pleasantly different. To eliminate the aliens, you have to dig a hole in the floor of the maze, and wait until one of them blunders into it. Then you have to frantically fill the hole in again before the beast can clamber out. The keyboard takes quite a hammering!

As well as having to avoid the monsters, and digging holes everywhere, you have another enemy to contend with: time. There is a three-minute limit on your achieving success, namely killing all of them off. This is not too bad when there's only four of them, but once you've successfully got rid of all of those, another six take their place to do battle *once* more. I've never got beyond this stage!

Scoring is achieved, as you might imagine, by actually succeeding in burying an alien. Points seem to be given out on a random basis, but are usually around 200 or 300. The faster you catch the aliens the more points you get.

The description of the game given to you hints at a monster worth 1000 points, but he seems very shy and doesn't come out very often. You have three lives in total. They go very rapidly at first, but you gradually begin to get the hang of things, and games last a little longer.

Summary
An original and interesting game well packaged and presented.

Kingdom of Nam

Microgame Simulations, 73 The Broadway, Grantchester, Cambridgeshire
ZX81 16K, cassette, price £4.95

'You are the ruler of Nam (sic) a small kingdom whose inhabitants are placid and industrious until they are misgoverned.'

If that introduction, from this program's accompanying sheet of paper, sounds familiar — well, it should. Although the concept appears under various names, it is a classic — one of the first interactive computer games developed, back in the Dark Ages of teletypes and Fortran.

The game is a semi-serious simulation. The user must plan allocation of resources year by year while under threat of various calamities, and not be 'deposed'.

Microgame's version works well in the main and can be gripping, although it is written in uninspired Basic and not entirely debugged, let alone idiot-proofed.

Without wishing to detract from the program's good points, here are some which could help the publishers to make the game better.

○ You need pencil and paper (or a good memory) while using Nam — your inputs do not appear on screen, so the user must check that they fit.
○ Data appears with up to five significant figures. What is 0.23575 of a factory, for instance?
○ Randomisation must not be overdone, otherwise real planning is impossible.

For example, in one run I was suddenly deposed in food riots although food stocks were huge.

Summary
Adequate, but not brilliant, this is still a gripping program. **KJ**

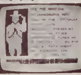
Monster Maze

J K Greye Software, 16 Park Street, Bath, Avon
ZX81 16K, cassette, price £5.95

Brilliant, brilliant, brilliant! Straightaway this gets into my personal top ten ZX programs.

Monster Maze is not entirely a novel concept, but it's very close to it. You are in an unknown maze and aim to get out — not novel, but here the screen displays what you would see, in superb 3-D.

A monster lurks in the maze to gobble you up — again not novel, but here, the monster is extraordinarily life-like and is quite frightening as it charges down the passage towards you. You can escape the monster by fleeing — yet again not novel.



Lastly, the inclusion of instructions is also not novel, but here those instructions are superbly written and scroll up half the screen past another superbly graphic creation, a lifelike, semi-animated bellman.

This is a fabulous program, written in Basic and machine code. I've only one criticism (and I'm not sure if it's a practical joke, or, conversely, whether there was an equipment fault). You're allowed to appeal after the monster gets you. When I won the appeal (which doesn't always happen), the program NEWed.

Summary
Undoubtedly one of the best ZX programs available.

Reviews
hardware

QS Hi-Res board

Quicksilva, 95 Brownhill Road, Maybush, Southampton
Price £35 Inc VAT

This printed circuit board fits into the QS or motherboard that provides edge connectors to give the ZX81 a screen resolution of 256 x 192 dots

This is greater than the first graphics mode on the BBC Microcomputer The letters and characters can be mixed with the graphics unlike some other computers

The board has no edge connector, so it has to plug into an edge connector, like the one on the 16K RAM pack A motherboard has edge connectors mounted in lines so that the boards can be mounted on them vertically as well as being able to connect the RAM pack or printer on the back to the ZX81

The screen is stored in 6K of RAM (6116s) and is controlled by a 2K ROM also mounted on the board This ROM contains all the routines for DRAWing, CLEARing the screen in black and white, PRINT strings, etc It also contains a BOX drawing routine as well as a self test program

The QS Hi-Res board

The Hi-Res screen replaces the Sinclair one, but it can be turned on and off under software control The commands are given in a REM statement following a line with the USR call to a routine in the ROM

Spaces must be used between commands, but the limit on the commands is only the line length of a REM statement — 256 characters, so you can have a whole graphics subroutine in one REM statement

It is very easy to use but all the variables have to be set before using the REM statement, which means a lot of lines defining the variables have to be written first The 6K of screen memory appears in the 32K 48K section meaning that you cannot use some of the larger RAM packs

The screen is sharp and because the commands are in ROM there is a great saving on RAM space as well as a massive increase in speed The screen is constantly in the 'slow mode even when plotting lines!

Summary
Very easy to use, the board offers a valuable extra dimension for ZX81 users **SA**

Vic Joysticks

Various suppliers and prices See below

If you have a Vic and the inclination to play games on it, you have no doubt run into the finger-knotting problems associated with keyboard control as most games require you to have the dexterity of a concert pianist with about the same size of hand

The answer is, as you reach for your wallet in frustration, a joystick! But which one? At present there are three different joysticks for the Vic here we take a look at all three

First Commodore's own joystick, available from any Commodore Vic dealer at a price of £7.95 including VAT The best way to describe its appearance is to think of an Atari joystick, change the logo, change the colour and *voila!* One Vic joystick

The only difference seems to be that the Commodore joystick is slightly chunkier in feel, and initially slightly stiff in movement, but this wears off after blasting down a few aliens It also appears to be slightly more robust, which is important with some of the more hectic games

The second joystick is of the proportional kind, and is manufactured by Stack Computer Services of Liverpool At a price of £14.95 including VAT it is a little more expensive, but the old adage of 'you pays your money must come into effect

This one is really of more interest to people writing their own software, as it doesn't employ the four joystick lines on the games port to feed its information to the Vic Instead it uses the POT X and POT Y lines, the result of which is that it will not run with any of the commercially available software I've seen

Finally we take a leap into the future with our third joystick, known simply a Le Stick. This one is currently available from the Vic Centre, at a price of £25 including VAT

This looks as if it would be more at home on the set of *Star Wars* rather than plugged into the back of the Vic!

The method of operation is very simple: contained in the joystick are mercury switches, which open or close depending on how the joystick is tilted This makes it very sensitive, and this is where my only real criticism lies in some cases it is too sensitive In other words if your hand is slightly off centre the joystick will register it accordingly and send your space ship careening off the screen

Summary
All three joysticks have their merits However, for pure games playing I think the size of your wallet will have to be the final answer **PG**

Explorer's Guide to the ZX81

By Mike Lord, published by Timedata, 120 pages paperback price £4 95

If I remind you that Timedata published the ZX80 and Acorn Atom *Magic Books*, you'll realise at once that the *Explorer's Guide* is of immediate interest and lasting value

I'm not sure why this isn't called the *ZX81 Magic Book*; it differs from its predecessors only in having more pages and fewer trivial programs All that extra space is taken up with a wealth of extremely useful hints, discourses and tips Whether you're a programmer, a hardware freak or just interested in using the ZX81, you're certain to find a lot of novel material here

The chapter on machine-coding is one of the best brief explanations of the subject I have come across

Best bits? I'd rather not vote, but I reckon a huge number of readers will particularly welcome these three hardware projects — a 1K pseudo-ROM interfaces for I/O (also described) and 16K RAM

Summary
Pretend it's your birthday and buy a copy **KJ**

Open Forum

Open Forum is for you to publish your programs and ideas. It is important that your programs are bug free before you send them in. We cannot test all of them. Contributions should be sent to: Popular Computing Weekly, Hobhouse Court, 19 Whitcomb Street, London WC2H 7HF.

Species — ZX81

This program, which is fascinating although very time-consuming to run, is a graphic example of the way computers can be used to simulate events in the real world.

"Species" pits two forms of life against each other in a closed environment, charting the growth and decline of the two species, one of which preys upon the other.

Species one is the weakest of the two, so when species one is plentiful, species two will increase rapidly in number, because of the plentiful supply of food.

However, there will come a point at which there are not enough of species one left to support the large population of species two, so species two will begin to decline.

At some point in the evolution of the two animals, the number of predators will have fallen so low that the prey species manages to multiply at a faster rate than that of the predator.

When you run the program, you will be asked "HOW MANY OF SPECIES ONE?" and then after answering this, "HOW MANY OF SPECIES TWO?". Enter your initial populations as a figure between one and five.

The program multiplies this by 10,000 to get the initial populations, and the populations of the two figures are continually updated on the screen as the graph of their relationship.

```
10 REM**SPECIES**
20 REM (C) MAHOGANY 1982
30 PRINT "HOW MANY OF SPECIES ONE?"
40 INPUT X
50 PRINT "OK ";X*10000;" OF SPECIES ONE"
60 PRINT
70 PRINT "HOW MANY OF SPECIES TWO?"
80 INPUT Y
90 CLS
100 FOR Z=1 TO 30
110 FOR T=1 TO 7
115 PRINT AT 1,1;"ONE> ";INT
    (10000*X+.5);" ";AT 19,22;"TWO> ";INT
    (10000*Y+.5);" "
120 LET X=X+(4*X-2*X*Y)*0.01
130 LET Y=Y+(X*Y-3*Y)*0.01
140 PLOT 7*X,7*Y
150 NEXT T
160 NEXT Z
```

YOUR PROGRAM COULD WIN A PRIZE

Each week from now on the editor will be going through all the programs that you send to Open Forum in order to find the Program of the Week.

The author of that program will qualify for DOUBLE the usual fee we pay for published programs, which is £10.

Then at the end of the month the four best programs of the week will go forward to our amazing Program of the Month contest, for which we are presenting a STAR prize.

This month the star prize is a super ZX printer, worth, £59.95!

And at the end of the year, all the best Programs of the Month will be entered in the super colossal competition, Program of the Year.

Sound like fun?

OK, GOTO your keyboard NOW!

Programs which are most likely to be considered for the Star Prize will be computer printed and accompanied by a cassette. The programs will be well documented, the documentation being typed with a double spacing between each line.

The documentation should start with a general description of the program and its functions and then give some detail of how the program has been constructed and of its special features.

Listings taken from a ZX Printer should be cut into convenient lengths and stuck down on to white paper.

Please enclose a stamped, self-addressed envelope.

Sketchpad — ZX81

This is a comprehensive sketchpad program for the ZX81. The O, W, E, A, D, Z, X and C keys are used to move the cursor around the screen in the following way:

Q	W	E
A		D
Z	X	C

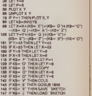
When the program is first run, the cursor is not plotting. Use "P" to start it plotting and "O" to stop again.

When the cursor is not plotting it can be used to run out mistakes. "O" resets the cursor to the bottom-left corner of the screen, and "V" clears the screen. "G" will send a copy of the picture to the printer.

Pressing "S" will save a picture on tape, using the name "SKETCH", and "J" will load a program from tape, and continue from where you left off. LOAD "SKETCH" may also be used to load a picture from outside the program.

Finally, "9" loads the picture into the string A$. PRINT A$ will then print the picture again. This could enable a picture created with this program to be used in conjunction with another program.

If the program is stopped, it can be continued with "GOTO 2000", as long as "9" was used before the program was halted.



```
10 REM "SUPER SKETCHPAD"
20 LET X=0
30 LET Y=0
40 LET P=0
50 PLOT X,Y
60 UNPLOT X,Y
70 IF P=1 THEN PLOT X,Y
80 LET K$=INKEY$
90 LET X=X+(K$="E")+(K$="D")+(K$="C")
   -(K$="Q")-(K$="A")-(K$="Z")
100 LET Y=Y+(K$="Q")+(K$="W")+
   (K$="E")-(K$="Z")-(K$="X")-(K$="C")
101 IF Y>43 THEN LET Y=43
102 IF X>63 THEN LET X=63
103 IF Y<0 THEN LET Y=0
104 IF X<0 THEN LET X=0
110 IF K$="P" THEN LET P=1
120 IF K$="O" THEN LET P=0
125 IF K$="G" THEN COPY
130 IF K$="O" THEN LET X=0
140 IF K$="O" THEN LET Y=0
142 IF K$="V" THEN CLS
145 IF K$="9" THEN GOSUB 1000
146 IF K$="S" THEN SAVE "SKETCH"
147 IF K$="J" THEN LOAD "SKETCH"
150 GOTO 50
1000 FAST
1001 DIM A$(704)
1005 LET O=PEEK 16396+256*PEEK 16397
1010 FOR I=0 TO 21
1020 FOR J=1 TO 32
1030 LET A$(J+32*I)=CHR$ PEEK (O+J+33*I)
1040 NEXT J
1050 NEXT I
1055 SLOW
1060 RETURN
2000 PRINT A$
2010 GOTO 50
```

Anagrams — ZX81

Clearly any process which will jumble the letters of a name is helpful with anagrams, in that it suggests new ideas. The problem is that every time you get the idea that a certain word can be extracted, you have to stop work out what is left, and start to work on the remainder.

This program saves you the trouble. Having RUN the program you simply enter the name — no spaces — from which you hope to construct an anagram. You will be faced with another prompt but the first time around it can be ignored by simply pressing NEWLINE.

The program will now print out 21 jumbled versions of the original string.

If none of them suggest anything to you, simply press NEWLINE and another 21 will be displayed. If, after one or more screenfuls you have some words in mind which can be formed from the original string, press the zero key at the end of the screen-filling process. The screen will clear and you will be faced with a prompt to enter a string.

If you now enter one or more words which you have seen can be formed from the original string, the letters of the word(s) you have entered will be subtracted from the original string and the program will get on with the job of jumbling what is left.

If, after one or more screenfuls, you decide that you are not going to complete an anagram with the particular word or words you chose taken out, press zero again and when faced with the string prompt simply press NEWLINE. This will result in the original string being restored.

In this way you can examine various options with ease.

Using the program I have found my own capacity to create anagrams has been much enhanced. So DENNY CLUE GROOVE OK — or good luck everyone.

Here are some notes on how the program works.

Line 10. CODE values and the VAL function are used throughout the program to replace literal numbers and save memory.

Line 20. The string from which you wish to form an anagram.

Line 40. D$ is the string, if any, which you wish to remove for the moment.

Anagrams
By David Lawrence

```
10 LET A CODE " $ (GRAPHIC 1
20 INPUT B$
30 LET C$=""
40 INPUT D$
50 FOR J=A TO LEN D$
60 FOR I A TO LEN B$
70 IF B$(I)<>D$(J) THEN GOTO VAL "100"
80 LET B$(I)
90 GOTO VAL "110"
100 NEXT I
110 NEXT J
120 FOR I-A TO LEN B$
130 IF B$(I)<>" " THEN LET C$=C$+B$(I)
140 NEXT I
150 PRINT ">" D$
160 FOR J=A TO CODE "+"
170 FOR I-A TO LEN C$
180 LET R-INT (RND*LEN C$+A)
190 LET T$ C$(R)
200 LET C$(R)-C$(I)
210 LET C$(I)-T$
220 NEXT I
230 PRINT C$
240 NEXT J
250 INPUT D$
260 CLS
270 IF D$<>"0" THEN GOTO
    CODE    (INVERSE "-")
280 LET B$ C$+D$
290 GOTO CODE "2"
```

Resistor colour codes
By David Lawrence

```
1 LET A  ODE "  (GRAPHIC 1
2 LET B-CODE " "(GRAPHIC 7)
3 LET C-CODE "  (GRAPHIC 5)
10 CLS
20 PRINT "VALUE"
30 INPUT N
40 PRINT N
50 IF N-(A<>A) THEN GOTO 130
60 LET P-INT (LN (N/C)/LN C)
70 LET V$=STR$ (N/(C**P))
80 PRINT C$(VAL V$(A)+B)
90 PRINT C$ VAL V$(A+A)+B)
100 PRINT C$(P+B)
110 INPUT D$
120 GOTO C
130 CLS
140 FOR I-A TO CODE "£"
150 PRINT I," " C$(I)
160 NEXT I
170 LET V$=""
180 FOR I=A TO B
190 PRINT "BAND" I;" ",
200 INPUT N
210 IF N-(A<>A) THEN GOTO 0
220 PRINT C$(N)
230 LET V$=V$+STR$ (N-B)
240 NEXT I
250 PRINT "VALUE "
    VAL V$(A TO A+A)*C**(VAL V$)B TO  )
260 INPUT D$
270 GOTO 130
1 SILVER
2 GOLD
3 BLACK
  BROWN
5 RED
  ORANGE
  YELLOW
  GREEN
  BLUE
10 VIOLET
11 GRAY
12 WHITE
BAND 1 GREEN
BAND 2 BLUE
BAND 3 RED
VALUE 5600
```

Line 50. These two loops remove the letters in D$ from B$.

Line 120. This loop copies B$, but without the spaces where letters have been removed.

Line 150. This loop exchanges each character in the string, in turn with another whose position is randomly chosen. Having done this the jumbled string is printed.

Line 250. A device to leave the screen display intact until NEWLINE is pressed.

Line 280. Any letters removed by the loop at 50 are reinstated if required.

```
10 LET A=CODE
20 INPUT B$
30 LET C$=""
40 INPUT C$
50 FOR J A TO LEN C$
60 FOR I A TO LEN B$
70 IF B$(I)<>D$(J) THEN GOTO VAL 100
80 LET B$(I)=""
90 GOTO VAL 110
100 NEXT I
110 NEXT J
120 FOR I=A TO LEN B$
130 IF B$(I)<>   THEN LET C$ C$+B$(I)
140 NEXT I
150 PRINT ">" ;D$
160 FOR J A TO CODE +
170 FOR I=A TO LEN C$
180 LET R=INT (RND*LEN C$+A)
190 LET T$=C$(R)
200 LET C$(R)=C$(I)
210 LET C$(I)=T$
220 NEXT I
230 PRINT C$
240 NEXT J
250 INPUT C$
260 CLS
270 IF C$<> 0 THEN GOTO TO CODE
280 LET B$ C$+D$
290 GOTO CODE "2
```

Resistor Colour Codes — ZX81

This 1K program for a ZX81 performs the dual function of interpreting resistor colour codes into numerical values and of constructing colour codes from numerical values.

Using the program is simple. Start up with GOTO, since there is an undeclared string matrix (C$) and the machine will request the input of a value.

To obtain an interpretation of the colour code, simply input the colours of the first three bands in response to the prompts and the value will be calculated.

The program does not deal with the fourth band which, if present, expresses the tolerance of the resistor.

In the event that you wish to return to the section of the program which constructs the correct colour code, simply input zero when asked to specify the colour of the first band.

Here are some notes about the program.

Lines 1, 2 and 3. Since the program must be started up with GOTO, there is no reason why A, B and C should not be input in direct mode, thus saving all the space allocated to these three lines CODE values are employed here to avoid the use of literal numbers and hence save memory

Line 50. (A<>A) is equal to zero and shorter in the computer's memory

Line 60. This obtains the integer value of N when expressed in logs to base 10

Line 70. Having reduced N to two figures it is transformed into a string thus making it a great deal easier to deal with the individual digits

Line 80 C$ is the undeclared matrix It must be dimensioned in direct mode (DIM C$(12,6)) and then filled with the following 12 colours, in this order SILVER, GOLD, BLACK, BROWN RED, ORANGE, YELLOW, GREEN, BLUE, VIOLET, GREY, WHITE

Line 250. The value of the resistor is calculated quite simply by constructing a two-figure number from the first two bands, then multiplying by a power of 10 represented by the third band.

Line 260. The input here and at line 110 are simply devices to leave the output of the program on the screen until you press NEWLINE

Telephone Book *Vic-20*

This program is written for an unexpanded Vic and acts as a storage and retrieval system for telephone numbers

It has a capacity of 25 names and numbers although this can be increased by changing the dimension value of TN$ and the value of TN in line 110

Numbers are input via option four and can then be saved to tale via option two If you want to access names or add more names to an existing list then go to option one which pulls the names off the tape

Lines 470 to 500 provide a matching search facility which means that if you can't remember the whole name of the person whose telephone number you want, you can just input a part of it and the routine will try and match it on to the names in its index and display all the relevant comparisons

Telephone book
By Dave Middleton

```
10 REM **************
20 REM * TELEPHONE  *
30 REM *            *
40 REM * BOOK       *
50 REM *            *
60 REM * C  PALMER  *
70 REM **************
80
90
100 REM START & SETUP
110 DIM TN$(25,1) CL=36879 TN=25 AN=1 POKE 36878,15
120 POKE CL,"9
130 PRINT "
140 PRINT "TELEPHONE  NUMBER"
150 PRINT "
160 PRINT "  1 = READ INDEX
170 PRINT "  2 = WRITE INDEX
180 PRINT "  3 = SEARCH NUMBER
190 PRINT "  4 = INPUT NUMBER
200 PRINT "  5 = END
210 PRINT "SELECT OPTION"
220 GETA$ IFA$="  "THEN220
230 A=VAL(A$) ONAGOSUB250,340,440,590,650
240 GOTO 120
250 REM READ
260 POKECL,8+17*5 PRINT "READING INDEX
270 OPEN 1,1,0
280 INPUT#1 AN
290 FORI=1TOAN-1
300 INPUT#1 TN$(I,0) TN$(I,1) NEXT I
310 CLOSE 1 FL=1
320 PRINT"READ COMPLETE !"
330 RETURN
340 REM WRITE
350 POKECL,8+17*7 PRINT "WRITING INDEX
360 OPEN 1,1,1
370 PRINT#1 AN
380 FORI=1TOAN
390 PRINT#1,TN$(I,0) PRINT#1,TN$(I,1) NEXT I
400 PRINT#1 "**********
410 CLOSE 1 FL=0
420 PRINT"WRITE COMPLETE !"
430 RETURN
440 REM SEARCH
450 POKE CL,8+17 PRINT " SEARCH NUMBER"
460 INPUT"NAME  - N$
470 FORI=1TOAN-1
480 FORT=1TOLEN(TN$(I,0))
490 IFMID$(TN$(I,0),T,LEN(N$))=N$THEN520
500 NEXTT,I
510 PRINT"NAME NOT FOUND  FORP=1TO2000 NEXT RETURN
520 PRINT"NAME  -" TN$(I,0)
530 PRINT "TEL NO  -" TN$(I,1) PRINT "CONTIUE (Y/N)"
540 GETA$ IFA$="THEN540
550 IFA$="Y"THENGOTO500
560 PRINT "HIT A KEY
570 GETA$ IFA$="THEN570
580 RETURN
590 REM INPUT
600 POKE CL,8+17*4 PRINT "INPUT NUMBER   AN
610 IF AN>TNTHENPRINT " NO SPACE LEFT !!" FORT=1TO1500 NEXT RETURN
620 INPUT "NAME  - TN$(AN,0)
630 INPUT "TEL NO  -" TN$(AN,1)
640 AN=AN+1 FL=1 RETURN
650 REM END
660 POKECL,8+17*2 PRINT " EXIT FROM PROGRAM"
670 IF FL=1THENPRINT "WARNING DATA NOT SAVED" GOTO 690
680 PRINT "   END
690 PRINT "   HIT A KEY
700 POKE36876,250 GETA$
710 POKE36876,220 IFA$=""THEN700
720 POKE36876,0 RETURN
```

Australia *Vic-20*

This small program shows what can be achieved by simple manipulation of the user defined character set

Lines 70 to 110 copy the character set from ROM to RAM but instead of just doing a straight copy it reverses the byte order inside each character

The result you can see when you run the program I think you will agree this could be quite a useful routine to be included in any program destined for our colonial shores

```
10 REM AUSTRALIA !!!
20 REM
30 REM CHRIS PALMER
40
50
60 PRINT"JABCDEFGHIJKLM
   NOPQRSTUVWXYZ"
70 FORT=0TO26*8STEP8
80 FORI=0TO7
90 POKE7168+I+T,
   PEEK(32760+T+7-I)
100 NEXTI
110 NEXTT
120 POKE36869,240
130 FORT=1TO500 NEXT
140 POKE36869,255
150 FORT=1TO500 NEXT
160 GOTO 120
```

Australia
By Chris Palmer

Space Pilot *ZX81*

This program puts you in command of a spaceship that must negotiate its way past some tricky obstacles to a safe landing at the local spaceport

To help you do this you are supplied with a graphic representation of the obstacles and of your ship's track together with a display of instruments which show altitude, speed of descent, position along the ground, speed across the ground, fuel remaining and, finally, the state of the 'command cycle'

This latter is a recurrent cycle, marked by the appearance of the figures 1 to 6 in the COM CYCLE position on the instrument panel during which any commands must be entered — once the cycle is ended the ship moves, the instruments are changed and the command cycle starts again

Commands available include three degrees of upward thrust, represented by the keys 1, 2 and 3 The thrust on key 1 will maintain the ship at the same rate of descent for the remainder of that cycle, 2 will reduce downward speed by 40 units 3 by 60 units

'Descent actually refers to movement across the screen in order to give a longer run in Speed 'along' the ground is, therefore, movement either up or down the screen and is controlled by the 6 and 7 keys which change the speed by 60 units

At the start of the game the obstacles are set up in a fresh configuration and the ship, a pixel point, is found located in the bottom left corner of the screen Obstacles are represented by grey blocks and the port for which the ship heads is represented by an inverse space set into the right-hand wall of the navigable area

The ship does not begin the game stationary it has a descent rate of 720 units (where 1000 units represents the width of one print position) and a speed in relation to the ground of 330 units (positive speed is speed up the screen, negative speed is speed down the screen)

First task is to brake the descent of the ship to prevent it crashing into the first barrier, which is some 5000 units below it

Successful piloting depends upon proper understanding, not only of the visual display of the ship s position, but of its instruments as well

When the ship is within 500 units of an obstacle it will, apparently, be touching it and no further movement towards the obstacle will be detectable on the visual display until the ship crashes

Using the instruments it is possible to keep tabs on the ship's position in relation to the obstacle and its speed

To land, you must hit the ground at less than 60 units, descent rate and less than 100 units ground speed— watch your ground speed and position as you come into land, you cannot set ground speed to zero, the minimum is 30 units either way

Apart from crashing, which naturally ends the game, time is limited by your consumption of fuel Main thrusts use 10, 30 and 50 units of fuel respectively while speed adjustments require 30 units

In addition, the ship uses five units of fuel each command cycle regardless of any use of thrust This prevents the pilot from setting ground speed close to zero all the way down and manoeuvring very slowly, since any gain in ease of control is offset by the fact that fuel is eaten up with very little movement

Once fuel is used up you have no further control over your descent

One final hint This is a leisurely game You will not be landing within a minute or so of starting If during times when you are coasting and not under pressure to take any decisions, you wish to speed up the game slightly, keep your finger on the NEWLINE key

This will terminate the command cycle on '1' and move the ship along more quickly

Here are some notes about the program

Line 120. This section down to line 320, sets up the playing area There are two barriers with random gateways and 15 random obstacles Spaceport position is set by lines 300-320

Line 360 This line and line 450 obtain the address of the byte before the display file

Line 370. These three lines and 460-480 translate the pixel point into an address in the display file in order that the program may check that the ship has not crashed into an obstacle

Line 419. These four lines use straightforward ballistic formulae to increment the position of the ship

Line 529. X represents acceleration due to gravity

Line 739. Note use of logical statements allowing what would otherwise have been three IF THEN LET statements to be combined on one line

Variables

S3 represents 1000 times the pixel x-coordinate for the ship
S4=1000 times pixel y-coordinate
U=descent velocity
T=time period for ballistic formulae
X=descent acceleration
X1=ground speed acceleration.
U1=ground speed
Note S3 is not the altitude of the ship — this is represented by 32000-INT(S3/2)-1000
Position is represented by INT(S4/2) 1000

PROGRAM OF THE WEEK

Space Pilot

By David Lawrence

```
1 GOTO 4
2 SAVE "PILOT"
3 STOP
4 RAND
10 REM ***********************
20 REM SET-UP
30 REM ***********************
40 LET FUEL=3500
50 LET S3=0
60 LET S4=2000
70 LET U=720
80 LET T=2
90 LET X=10
100 LET X1=0
110 LET U1=330
120 PRINT AT 4,0,"32 x GRAPHICS A"
130 PRINT AT 21,0;"32 x GRAPHICS A"
140 FOR I=4 TO 20
150 PRINT AT I,7,"GRAPHICS A"
160 PRINT AT I,16,"GRAPHICS A"
170 IF I<13 THEN GOTO 190
180 PRINT AT 14,I+10,"GRAPHICS A"
190 PRINT AT I,31,"GRAPHICS A"
200 NEXT I
210 FOR I=1 TO 15
220 LET R1=INT (RND*30)+1
230 LET R2=INT (RND*16)+5
240 PRINT AT R2,R1,"GRAPHICS A"
250 NEXT I
260 LET R=INT (RND*15)+5
270 PRINT AT R,7,"SINGLE SPACE",
    AT R+1,16,"SINGLE SPACE"
280 LET R=INT (RND*15)+5
290 PRINT AT R,16,"SINGLE SPACE",
    AT R+1,16,"SINGLE SPACE"
300 LET R=INT (RND*16)+5
310 IF R=14 THEN GOTO 300
320 PRINT AT R,30,"SPACE GRAPHIC SPACE"
330 REM ***********************
340 REM POSITION CALCULATIONS
350 REM ***********************
360 LET P=PEEK 16396+256*PEEK 16397
370 LET P1=INT ((S3/1000)/2)
380 LET P2=21-INT ((S4/1000)/2)
390 LET P3=PEEK (P+P1+33*P2+1)
400 GOSUB 530
410 LET S3=S3+U*2
420 LET U=U+X*T
430 LET S4=S4+U1*2
440 LET U1=U1+X1*T
450 LET P=PEEK 16396+256*PEEK 16397
460 LET P1=INT ((S3/1000)/2)
470 LET P2=21-INT ((S4/1000)/2)
480 LET P3=PEEK (P+P1+33*P2+1)
490 GOTO 360
491 REM
492 REM
493 REM
494 REM
495 REM
500 REM ***********************
510 REM INSTRUMENTATION
520 REM ***********************
530 LET X=10
540 LET X1=0
550 PLOT INT (S3/1000),INT (S4/1000)
560 PRINT AT 0 0,
570 PRINT "ALTITUDE ";32000-INT (S3/2)-1000,' ",
580 PRINT "DESCENT.",INT U," ",
590 PRINT "POSITION:";INT (S4/2)-1000;" ",
600 PRINT "SPEED ",INT U1;" "
610 PRINT AT 3 0,"FUEL ";FUEL," "
620 IF P3=136 OR P3=128 THEN GOSUB 800
630 LET FUEL=FUEL-5
640 PRINT AT 3,16,"COM.CYCLE."
650 FOR J=1 TO 6
660 IF FUEL<=0 THEN GOTO 710
670 PRINT AT 3 27;J
680 IF INKEY$<>"" THEN GOTO 720
690 NEXT J
700 PRINT AT 3,27,"SINGLE SPACE"
710 RETURN
720 LET X1=(INKEY$="7")*30-(INKEY$="G")*30
730 LET X=10=INKEY$="3")*50-(INKEY$="2")*30
    -(INKEY$="1")*10
740 IF FUEL<=0 THEN GOTO 760
750 LET FUEL=FUEL-ABS X1+(X-10)
760 RETURN
770 REM ***********************
780 REM CRASH OR LANDING
790 REM ***********************
800 IF P3<>136 THEN GOTO 830
810 PRINT AT 10,10;"YOU HAVE CRASHED"
820 STOP
830 IF INT (S3/1000)<62 THEN RETURN
840 IF U1<=100 AND U<=60 THEN
    PRINT AT 10,10,"CONGRATULATIONS"
850 IF ABS U1>100 OR U>60 THEN
    PRINT AT 10,10,"SORRY, TOO FAST"
860 STOP
```

Line graph
by Eric Deeson

```
 7 FOR B=1 TO [illegible]
 8 INPUT X(B)
11 [illegible]
1? NEXT B
1? CLS
17 FOR B=0 TO [illegible]
1? LET M=[illegible]
19 LET [illegible]
20 FOR [illegible] TO [illegible]
2? LET [illegible]
22 PLOT [illegible]
23 NEXT [illegible]
24 NEXT B
```

To use, pre-enter LET [illegible] [illegible] to [illegible] the number of [illegible] and enter DIM [illegible] and [illegible]

Line graph ***ZX81***

Graphs bore some people, and graph-plotting programs are hardly news. But a program which plots input points and joins them with straight lines is useful, and such a routine for the **ZX81** doesn't seem to be well known.

This program is for both 1K and 16K memory sizes. The version listed works fine in 1K. Omitted lines are for 16K messages and programmed (rather than direct) assignments. In 16K one can of course polish the program more, to print title, axes and axes graduations.

Note that as it stands, X must be in the range 0 to 63, and Y in the range 0 to 43.

Where is Venus? ***ZX81***

If the amateur astronomer wishes to look for a particular star or planet he must first obtain the right ascension and declination from some source such as *Whitaker's Almanack*.

He then requires to know his own latitude and longitude and the Greenwich Mean Time for the day in question. When this information is fed to the ZX81 this program will calculate the local sidereal time (star time) and the altitude and azimuth of the object.

If the altitude comes out negative, the object is below the horizon!

When inserting time in hours and minutes, or degrees and minutes, remember that 6 minutes = 0.1, eg, 11h12m = 11.2 hrs, and 12°18' = 12.3°.

The program as written is for Bolton, Lancashire. In lines 330 and 370 replace the latitude of Bolton by your own latitude, and in line 270 insert −0.066 for every degree you are west of Greenwich.

Where is Venus?
by William Cartwright

```
5 REM "DATE TO DAYS, L.S.T., ALT AND AZ"
6 PRINT "NAME OF STAR OR PLANET"
7 INPUT A$
8 PRINT A$
10 PRINT "INPUT MONTH, DAY, HOUR"
15 PRINT
20 INPUT M
25 PRINT "MONTH ";M
26 PRINT
30 INPUT D
35 PRINT "DAY ";D
36 PRINT
40 INPUT T
45 PRINT "TIME ";T
46 PRINT
50 IF M>2 THEN GOTO 80
55 LET B=M-1
60 LET R=INT (B*63/2)
65 LET P=R+D
70 GOTO 120
80 LET A=M+1
90 LET C=INT (A*306/10)
91 PRINT
100 LET H=C-63
110 LET P=H+D
120 PRINT "DAYS ";P
121 PRINT
210 REM "GMT TO L.S.T."
220 LET A=P*0.065709
240 LET E=A-17.377592
260 LET S=T*1.002743
270 LET V=S+E-.166
280 IF V>24 THEN LET V=V-24
281 IF V<0 THEN LET V=V+24
290 PRINT "LST ";V
291 PRINT
292 PRINT "INPUT DEC"
300 INPUT DEC
305 PRINT "DEC ";DEC
306 PRINT
307 PRINT "INPUT RA"
310 INPUT RA
315 PRINT "RA ";RA
316 PRINT
320 LET H=(V-RA)*15
330 LET ALT=(SIN (DEC*PI/180)*SIN (53.5*PI/180))+(COS (DEC*PI/180)*COS (53.5*PI/180)*COS (H*PI/180))
340 LET O=ASN ALT*180/PI
350 PRINT "ALT = ";O
360 PRINT
370 LET AZ=(SIN (DEC*PI/180)-SIN (53.5*PI/180)*SIN (O*PI/180))/(COS (53.5*PI/180)*COS (O*PI/180))
380 LET Z=(ACS AZ)*180/PI
390 LET W=SIN (H*PI/180)
400 IF W<0 THEN PRINT "AZ = ";Z
410 IF W>0 THEN PRINT "AZ = ";360-Z
420 STOP
```

Sound & vision



Good tunes do from little Acorns grow

The musical possibilities of the Atom are fairly limited but in their way they serve to show the basic principles of computer music in a very clear way

The really good thing about the way music can be made on the Atom, however, has nothing to do with the hardware, but is a consequence of the machine's software

In fact the flexibility of the loudspeaker output — from the point of view of programming it — makes a crude music generator into a very useful one

The beauty of the Atom is the assembler, which resides in the Basic It isn't particularly unusual to find assemblers for micros but it is very unusual to find an assembler in ROM, easily accessible from Basic on a low price machine

What's so special about an assembler, and why mention it in a music column? First, programs written in assembly code are converted directly to the binary machine code which is the actual language used by the microprocessor

Music is made normally by sound waves, which are vibrations of the air causing a sensation in the ear The frequency of the vibration determines the pitch of the note heard, the higher the frequency, the higher the pitch Usually music is made of many different pitches 'bound-together'

The frequency of the vibration can also be thought of as a speed, which relates to the speed of the computer,

The speed at which bits are output to the loudspeaker on the Atom is the speed at which the loudspeaker disturbs the air, and consequently the frequency that is heard by the ear The highest frequency that can be output at the loudspeaker from normal Basic is not very high

In the manual it is explained that the address of the output port for the loudspeaker is hexadecimal B002 A note of 187 Hertz — which is somewhere near the lower end of the conventional music scale — can be generated by the short program

```
10 P = #B002
20a ?P = ?P 4 GOTO a
```

To tune this to an actual note — which, incidentally, must be F# or lower — extra dummy commands must be inserted to slow the looping down to that frequency Unfortunately this procedure is not easy and the notes that can be obtained in this way are not often exact

The difficulties encountered in getting the Atom to play exact notes from Atom Basic show why it is so important to be able to use and assemble when writing computer music software The speed of an assembler becomes especially important when the music is played through a loudspeaker connected directly to an output port

If any readers have managed to obtain an evenly-tempered scale — however approximate — on the Atom, please write to me at *Popular Computing Weekly*, Hobhouse Court, 19 Whitcomb Street, London WC2 7HF I will print the best ideas **Sam Blythe**

How to draw a line from A to B

The one I wrote is called *Twixt*, in North America it's *Tween* Doubtless there's one called *Splurge* somewhere

I'm referring to programs that do 'in-betweening' You give the computer one shape, then another, and it calculates all the shapes that lie in between the two extremes — as many or as few as you want to have

There are many complex ways of doing this, but the program shown here has the virtue of simplicity, and thus ease of control The most advanced kinds of computer animation using in-betweening are right at the leading edge of what is possible with computers Most of the images you see on tv screens in computer-aided adverts are made not in 'real time', but one frame at a time When run rapidly, the sequence gives the illusion of gradual movement

Our program, however, moves in real time with very simple shapes which need less computing But don't bank on making any animation artist redundant with it.

A glance at the program — written for the BBC Micro model A or B — will probably show how it works Imaginary straight lines are drawn between the first and last shapes, and these lines are divided into equal segments These points are then joined up

```
10 REM INBETWEENING FOR BBC MICRO
20 REM N.B. INITIAL IMAGE IS NOT DRAWN OUT
30 REM CAN YOU SEE HOW TO CORRECT THAT?
40 MODE4:CLG
50 INPUT NO. OF POINTS (LESS THAN 10) P
60 FOR I=1 TO 2:PRINT IMAGE  I
70 FOR N=1 TO P
80 PRINT POINT  N:INPUTX(I,N) Y(I,N)
90 REM TYPE IN POINT AS X,Y
100 NEXT N:NEXT
110 INPUT HOW MANY STEPS (<=)
    S:IF S>10 THEN110
120 CLS:FOR I=1 TO S
    FOR F=0 TO ...ST ST 1:S:NEXT F
130 FOR J=1 TO P
140 A=X(1,J)+(ST*(X(2,J)-X(1,J)))
    B=Y(1,J)+(ST*(Y(2,J)-Y(1,J)))
150 IF J=1 THEN MOVE A,B ELSE DRAW A,B
160 NEXT J:ST  NEXT I
```

The program should easily adapt to other makes of computer All you need is the ability to plot points and draw lines No points are necessary as far as the images are concerned, but in computer graphics you frequently have to give a line a point to start from, rather than just trailing across from the last line that you drew

Finally, can you imagine any other sorts of in-betweening that you could perform on a BBC micro? Music, for example? **Brian Reffin Smith**

MORE FUN WITH YOUR ZX81!



The addition of our fully-compatible, assembled, tested and guaranteed

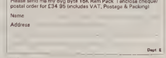
16K RAM PACK

PUSH-ON PLUG-COMPATIBLE MEMORY

Means more memory for better games and bigger programs

ORDER YOURS TODAY FROM:

CAPS LTD. Dept. E, 28 The Spain, Petersfield, Hants GU32 3LA


Please send me my Byg Byte 16K Ram Pack. I enclose cheque/postal order for £34.95 (includes VAT, Postage & Packing)

Name

Address

Dept E

Allow 28 days for delivery

Fully inclusive price

£34.95

Hand & mouth



The billiard ball program style

If you want to build a program out of modules, you should aim to leave the central processor unit (CPU) in the same state at the end of the module as it was at the beginning.

The 6502 CPU has three general purpose registers and the processor status register (PSR). The instructions to save all the registers are.

```
PHP    Push PSR
PHA    Push Accumulator (A)
TXA    Transfer X to A
PHA    Push A
TYA    Transfer Y to A
PHA    Push A
```

When you 'Push' a register, it is saved in a particular place in the memory where you can get at it later on, so, at the end of the subroutine (SR) or module you need the following instructions

```
PLA    Pull A
TAY    Transfer A to Y
PLA    Pull A
TAX    Transfer A to X
PLA    Pull A
PLP    Pull PSR
```

Note that these instructions pull the registers out of memory in the reverse order. It's like rolling billiard balls into a closed tube. If you put the red in first followed by the green and the white you will get the white out first, then the green and finally the red.

A module that is encapsulated by the instructions to save and retrieve the registers really is transparent to the main program because the CPU looks just the same after as it did before. What has changed is the input/output from the program, and any storage locations used by the module.

Use a module as you fancy

The great benefit of this is that a subroutine or module can be used anywhere and will do only what you want.

If you start to construct a program by building subroutines to carry out the smallest tasks — for example to clear the screen of the VDU, increment a pointer in memory, output a character to a printer — then you can combine those SRs into more powerful modules.

Finally you will have a single DO loop that is the entry to the program where it waits for your command before diving off into lower level SRs.

The concept of linked subroutines with the addition of a common stack for transferring numbers and variables, is essentially the principle underlying Threaded Interpretive Languages (TILs).

Accepting the speed penalty

The main difference between the two is that an assembled program is dedicated to a particular purpose and can be more efficient in terms of speed and memory requirements while a program written in a TIL such as Forth or Stoic can be modified more easily.

Nevertheless, a compiled Forth program should look very similar to assembled object code and with the facility in some versions of Forth to compile only the core words used by a program may be of similar length.

The Z80 CPU has similar instructions to push the CPU registers.

If the speed penalty imposed by the time taken to get the registers in and out of memory is acceptable, you will find the construction of long programs far easier when each of your subroutines is sealed and usable anywhere in a program. **John Dawson**

The root cause of a sore forefinger

What is the easiest way of finding the square root of a number? There's no doubt that it involves the envy of the schoolboy of 20 or even 10 years ago, the omnipotent pocket calculator.

Equation solving, which sometimes took hours and ended with a pair of bloodshot eyes, can now be accomplished quickly and with little more than a sore forefinger. We have been released from the drudgery of the log book and slide rule!

But have you ever stopped and wondered how your calculator generates all of its standard functions. Clearly by pressing the square root button for instance we set in motion a microprogram which is permanently written in the calculator memory — ROM — and which is designed to generate a square root efficiently and accurately.

The rough overall procedure used in Hewlett-Packard and most other calculators to generate $\sqrt{X}$ is as follows

1. Guess an answer A.
2. Generate A^2
3. Calculate the error $E=X-A^2$
4. If E is less than the required error then $A=\sqrt{X}$
5. Depending on whether A is too large or too small, modify A and return to step 2

Obviously it is the algorithm implied in step 5 which will determine the utility of the method. This algorithm must be geared towards the Binary Coded Decimal arithmetic of the machine's microprocessor.

I will leave the essential constituent moves of step 5 until next week. For now I'll give you two points.

In general a number is stored in an HP calculator as a 12 digit mantissa between 1_{10} and 10_{10} with an exponent. The task of the rooting algorithm is to find the root of both. If we can engineer an even exponent every time then all we need do is to halve it and hence concentrate on the mantissa. Second, as we get closer to the true value of the root the value of the remainder R decreases. How do we maintain accuracy?

Competition

We are offering a prize of £5 for the most interesting program to generate a square root from first principles as above. **John Gowrie**

Programming

Some days 'ee chains his programs together

***Barry Cornhill** tells you how to run several programs one after the other by 'chaining'*

EVERY time you load a new program into your ZX81, the previous one is overwritten. However, anything placed above RAM top is not affected by the loading process

The following 'suite' of programs demonstrates how you can transfer data from one program to the next The demo programs are very short all you need to do is expand the DIM array to suit. In doing so you will need to increase the bytes to Poke My examples use only 56

To calculate the bytes required, decide on your array size and do the following calculation

```
4+2*no of dimensions +5*no of elements
eg B(10) = 4+2*1+5*10 = 56
B(2,10) = 4+2*2+5*20 = 108 etc.
```

Then calculate what to poke RAM to with

```
POKE 16389 128-INT (N/256)
```

where N = number of bytes to reserve

The first thing to do is type in the six programs as written, and save them individually on tape. the REM statements at the beginning of each program are the title 'names of the programs concerned

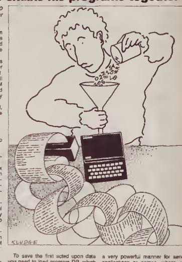
Note when typing in the last program (DP) enter the line 130 in normal video This will change automatically on subsequent re-loadings for auto running

The first step in running the programs and in all subsequent re-runs on switch-on is to reserve space in RAM top by typing as a direct command

```
POKE 16389 127 new line new line
```

Load program P1 This purely creates 10 simple numbers, and produces the same numbers every time, and places them in the array B(10) This must be the same for all programs concerned

Load and run, in any order programs P2, P3, P5 and P6 You will see that P2 and P3 act upon the data stored without altering the last original data, whereas P5 and P6 act upon and change the original data



To save the first acted upon data you need to load program DP, which will automatically save your data in a 'data file

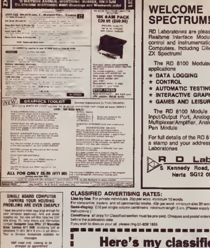
On a subsequent 'switch-on' POKE 16389. 127 new line NEW new line, load datafile It will automatically run and display the last lot of data. Loading programs P2, P3, P5 or P6 will now, as above, act upon this data.

You can then create a second data file (or as many as you like) or overwrite your original data file Program P1 is now of course redundant.

As you can see, this can be used in a very powerful manner for serious applications or games, where, with only 16K memory, you can in fact have a 15K file of data and say 10 1K programs to act on the data, giving an effective 25K of program length for example

Once the long wait in loading say the 15K datafile is over all subsequent 1K loadings (or whatever) will take only the time required to load a 1K (or whatever) program

I am currently looking into the technique of handling string data in the same manner



CLASSIFIED ADVERTISING RATES:

Line by line For private individuals 20p per word minimum 10 words.
For companies traders and all commercial trades 40p per word minimum also 20 words
Semi-display £10 per single column centimetre minimum length 3 cm. (Please supply A/W as PMT Or supply rough setting instructions.)
Conditions all copy for Classified section must be pre-paid. Cheques and postal orders should arrive at least two weeks before the publication date
If you wish to discuss your ad please ring 01-839 1855

Here's my classified ad.

(Please write your copy in capitel letters on the lines below.)

I make this ______ words at ______ per word so I owe you £ ______

Name ....
Address ....
....
....
Telephone ....

Please cut out and send this form to Classified Department *Popular Computing Weekly* Hobhouse Court, 19 Whitcomb Street, London WC2

Peek & poke

Peek your technical queries to Ian Beardsmore. He will poke back an answer.

HAVE YOU TRIED A TIN OPENER?

Peter McIntosh of Arcadia Road Glasgow writes

Q My ZX81 keeps crashing because my power supply jack is very loose in the socket. I thought you could unscrew the ZX81, and push the socket clips in to make them tighter But I cannot undo my ZX81 Evea after I have taken the three screws out the case stays together It is very annoying. I bought my computer ready made, can this only be done on kit versions?

A All ZX81's can be easily undone There are in fact *six* screws holding your case together, the problem is that three of them are hidden under the feet These foam squares are sticky backed and can easily be peeled away to reveal the hidden screws! When the case is undone you will see the PCB held on by two more screws once these are undone the jack socket is easily accessible The only care you have to take is to ensure that the ribbon cable between the PCB and the keyboard is not torn It is, however, quite a simple job, and after you have finished the feet just stick back down in their original places

ALL MY OWN WORK, WELL ALMOST DEPT.

Kevin Dyer of Langton Road, St Ives writes

Q I sometimes make changes in programs that I see in magazines or books, expecially if they are only 1K, because I have a RAM Pack Am I allowed to send these programs to a magazine to be published?

A This apparently simple question touches a grey area of the copyright law It all depends on what you mean by changes It could mean that you have added a subroutine, or routines which enhance the program, or it could mean that you have just changed some variable names and perhaps the occasional print statement I am sure — giving you the benefit of the doubt — that you are referring to the first case, but I shall try and explain both cases

If you have written a subroutine that perhaps improves the graphics of a program, or adds a scoring mechanism, then that subroutine is yours, although it is clearly of little use without the original If you wanted to use it you would have to obtain permission from the writer of the original program to use that program

It would make little sense to send up a subroutine without quoting the program it is meant to go with! (Though you might be allowed to get away with it, if you were not paid)

In a more practical light, you may well find that no one would want your subroutine anyway A lot of people add new features to published programs, but only use the new versions privately This is quite allowable But, if you want to send a subroutine to a magazine, get in touch with the original author first

The second type of change is much more serious, because it is almost certainly done with the aim of circumventing the law to make quick money at someone else's expense In theory a minor change is all that is needed to change copyright, but in two instances that I know about where small changes were made, and then the programs were offered for sale the offending software was withdrawn when legal action was threatened

Given that the bulk of law in this field is based on precedence, more than cosmetic changes would probably be needed to effect copyright

AN OLD 405 TELLY JUST WON'T TALLY

I B Fowler of Field Way Port Talbot writes

Q I am facing a problem I am sure that many computer enthusiasts face — that of negotiating television time in a household where computers are as big a mystery as the Bermuda triangle, and as remote as the outer solar system Somewhere up in the loft I have an old black and white television Is it worth my trouble getting this down That is, for the sake of peace will I be able to use it for my ZX81?

A Whether you can use it or not depends on whether it is a 625-, or a 405-, line television If it is 405 lines then the answer is no The frequency modulator in the ZX81 is set for 625 lines

SHOW ME THE WAY TO GO ROM

Simon Hennessy of Water Lane, Burton-on-Trent writes

Q How can you find out if you have a faulty ROM? I bought my ZX81 second hand and I think it is quite old If it is wrong then what do I do about it?

A There are a lot of tests that can be done, but try these two

```
10 PRINT SQR .25
        (The answer should be 0.5)
                  or
10 PRINT 2**32-1
(The answer should be 4294967295)
```

If you get the wrong answer to either of these then you have a faulty ROM

What you should do if this happens is get in touch with Sinclair's mailing company at this address Sinclair Research Ltd (JRS), Stanhope Road, Camberley, Surrey GU15 3PS Telephone Camberley (0267) 21282

IT ALL WENT BLACK 'N' ME MEMORY WENT

Ben Kellett of Cheyne Walk, Chelsea writes

Q I have just got my Sinclair RAM pack, after a wait of six weeks But I am getting a lot of problems with it Every so often the screen just blacks out and I am left with nothing

Have you any advice on what I can do to stop this problem as I do not want to risk a long delay by sending it back

A The 16K RAM pack is a major cause of problems, and you are no exception Most probably your trouble stems from the poor connection between the I/O port and the pack What can you do? If you return your pack the chances are there will be no improvement in the one you get back, because the source of the trouble is a basic design fault Try the following

1. Place a piece of card or hardboard under the pack This stops the pack 'hanging' on the port Even better, glue a strip of rubber on the underside of the pack to stop it slipping as well

2. When you have found a position where the pack works well, secure it there with a large sausage of Blu-Tack, between the pack and the case

3. Clean the contacts on the PCB

4. Lastly, but importantly, always use your computer on a hard flat surface

CAN I TAKE IT OVER TO DOWN UNDER?

Isaac Thompson of East Village Road Solihull writes

Q Our family is soon going to emigrate to New Zealand, and I would like to know if I can take my ZX81 with me and use it over there, or will I have to buy a new and different one?

A You will have to buy a new ZX81 which is compatible with the New Zealand television system It will in fact be the same as one you would buy anywhere else, apart from the frequency modulator, which controls the signal going to the television set As yet there are not a lot of ZX81s over there, but more are in the pipeline

The New Zealand distributor is David Reid Electronic Ltd, 3-5 Auburn Street, Ta Ka Puna, Auckland, New Zealand They should be able to supply you with one, or else put you in touch with your nearest dealer

Send your questions to Peek & poke *Popular Computing Weekly*, Hobhouse Court 19 Whitcomb Street London WC2 7HF

Competitions

1 Solve the puzzle and win £10!

In this remarkable cryptarithm, which was devised in the 1940s by Joseph Ellis Trevor of Cornell University, all the digits are prime

```
  * * * x (times)
    * *
  -------
  * * * *
* * * *
---------
* * * * *
---------
```

Can you write a simple program, and, using only the digits 2, 3, 5 and 7, find the missing numbers?

Maximum capacity problem: solution

From the diagram we can see that the volume of the tank, for a given value of X can be found from the formula

Volume = (10 – 2 * X) * (10 – 2 * X) * X

In solving this puzzle it is logical to assume (as is indeed the case), that in progressing from the value when X is very small, to the value when it is at a maximum, (5), the volume gradually increases to a maximum point before beginning to get smaller

In the program given, the value of X is set at a minimum, the volume worked out, and this volume is compared with the preceding volume to see if it is either equal or less This will mark the turning point.

In order to verify that there is only one maximum value the procedure can be reversed In this case a high starting point for X is given in line 10.

```
10 LET X = 5 – 0005
```

and this value is *decremented* in line 60

```
60 LET X = X – 0005
```

The results show that the answer lies between 1 667 and 1 666

After running the program through once, then the starting value of X can be re-defined as 1 666 and the steps by which it is incremented can be made smaller

The answer is in fact 1.6 recurring

Suggested program:

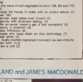
```
10 LET X = 0 0005
20 LET Y = 0
30 LET V = (10 – 2 * X) * (10 – 2 * X) * X
40 IF V < = Y THEN GOTO 100
50 LET Y = V
60 LET X = X + 0005
70 GOTO 30

100 PRINT X
110 STOP
```

Closing date for both the crossword and the puzzle is the Monday three weeks after the cover date

Please mark your envelope CROSSWORD or PUZZLE

2 Complete the crossword and win a gift voucher!

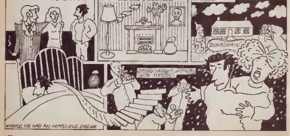


ACROSS
4 Basic statement to replace a vessel (5)
7 Basic word for penny is perform (7)
8 Path for the current church ship (6)
10 The best part of the group (5)
12 Made to return cheese (4)
13 Exerts quietly inserted by specialists (7)
14 God in ancient horror movie (4)
15 Greek character is formed by a thousand million million amps (6)
19 PS Leap about if you want to get some home computers (5)
20 The united nations, committed to be free (7)
21 Retain information the way lions does (5)

DOWN
1 A tiny piece of root vegetable found in 16A, 8D and 17D (6)
2 Keeps the house in order with an output device (7)
3 See 18D
5 Monster duck takes over church (3)
6 Operate with American energy (3)
8 The computer maker's got a foot complaint! (6)
9 Specify a condition (8)
11 Sees infiltrators (8)
12 People who tease slider on chip technology (7)
17 Deer taken (3)
18, 3 Audio interface byproduct involved in se-see arms (to start with) (3,3)
19 I am a unit (3)

CITIZEN PAIN

BY DAVID IRELAND and JAMES MACDONALD




EXPAND
YOUR
VIC
OFFICIALLY APPROVED BY COMMODORE
From £85 plus VAT.
ARFON PRINTER
A low cost stand alone printer which will be almost essential for your larger programs will be launched in the Spring of 1982. The power plug for this unit is already on your expanded system
EXPANDABLE TO 7 CARTRIDGES
You will now be able to use up to seven cartridges to expand from your basic Vic 20 These can include RAM memory expansion up to nearly 30K of usable memory ROM cartridges with packaged programs user expansion cartridges printer software disc software RS232, IEEE interface line expansion firmware and many others – besides of course all your games cartridges
ARFON EXPANSION MEMORY
Immediately available from Arfon in cartridge are 3K RAM +2 sockets 8K RAM 16K RAM 8K ROM 16K ROM Also a basic Vic simulator cartridge to allow tape and cartridge use without altering the system
VIC 20 Expansion System
The Arfon Vic 20 Expansion System is a finished metal cased unit that integrates your Vic 20 with an expansion board for up to seven cartridges and a toroidal power supply (fully enclosed) to give you sufficient power for any expansion and also to power the Arfon Vic Printer Your Vic and its screen modulator are incorporated into the expansion system to produce one unit and there is an optional lid that covers the expansion area and allows your TV to sit on top Access to the various input sockets on your basic Vic 20 is not restricted while it is in the expansion unit.
Stocks are available now through your own Vic Dealer
INTERFACES
Slots have been left to allow normal use of the cassette socket disc socket etc which will still run normally with your expanded system.
I would like to be included on your mailing list
Name
Address
Send to
Arfon Microelectronics Ltd
Cibyn Ind. Estate
Caernarfon Gwynedd
N Wales
Arfon Micro
AM
Write for further details